Implementing the AsyncTaskBarrier Framework Using Project Reactor (Part 1)

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the API of the AsyncTaskBarrier class for Project Reactor

Class AsyncTaskBarrier

```
public class AsyncTaskBarrier
extends java.lang.Object
```

This class asynchronously runs tasks that use the Project Reactor framework and ensures that the calling method doesn't exit until all asynchronous task processing is completed.

Method Summary

All Methods | Static Methods | Concrete Methods

Modifier and Type	Method	Description
`static void`	`register (java.util.function.Supplier<reactor.core.publisher.Mono<java.lang.Void>> task)`	Register the `task` task so that it will be run asynchronously when `runTasks()` is called.
`static reactor.core.publisher.Mono<java.lang.Long>`	`runTasks()`	Run all the register tasks.

See Reactive/flux/ex4/src/main/java/utils/AsyncTaskBarrier.java

Learning Objectives in this Part of the Lesson

- Understand the API of the AsyncTaskBarrier class for Project Reactor
- Know how to use AsyncTaskBarrier in practice

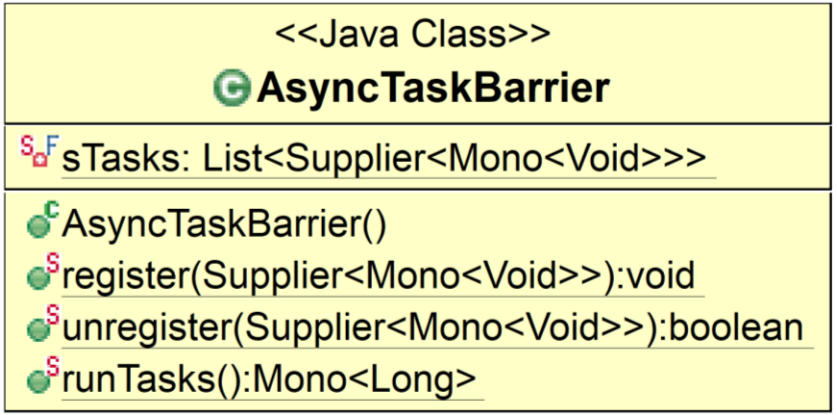
```
AsyncTaskBarrier.register(this::syncThrowException);
AsyncTaskBarrier.register(this::asyncThrowException);
AsyncTaskBarrier.register(this::syncNoException);
AsyncTaskBarrier.register(this::asyncNoException);

long testCount = AsyncTaskBarrier
     .runTasks()
     .blockingGet();

assertEquals(testCount, 2);
```

See Reactive/flux/ex4/src/test/java/utils/AsyncTaskBarrierTests.java

The AsyncTask Barrier Class API

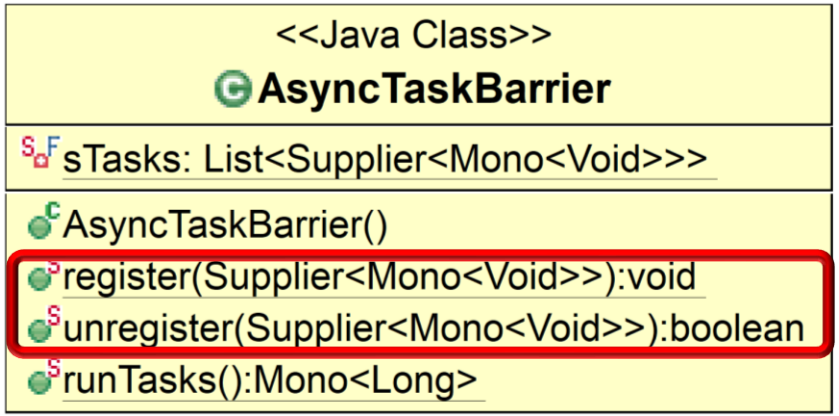
The AsyncTaskBarrier Class API

- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks



The AsyncTaskBarrier Class API

- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks
  - It provides methods that register & unregister tasks passed as Suppliers



Interface Supplier<T>

Type Parameters:

T - the type of results supplied by this supplier

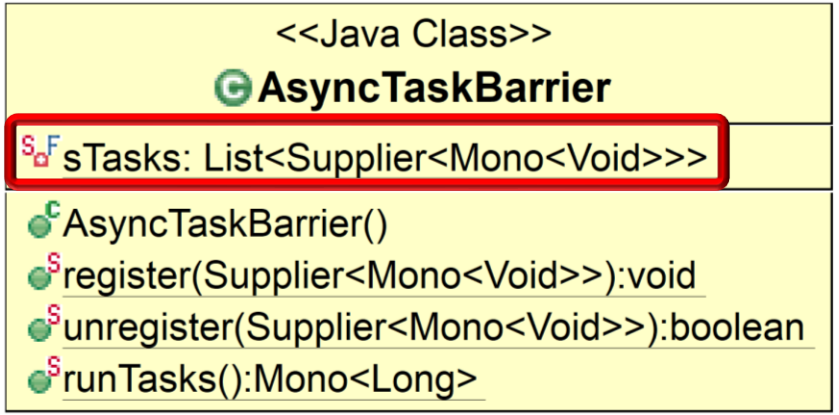
Functional Interface:

This is a functional interface and can therefore be used as the assignment target for a lambda expression or method reference.

See docs.oracle.com/javase/8/docs/api/java/util/function/Supplier.html

The AsyncTaskBarrier Class API

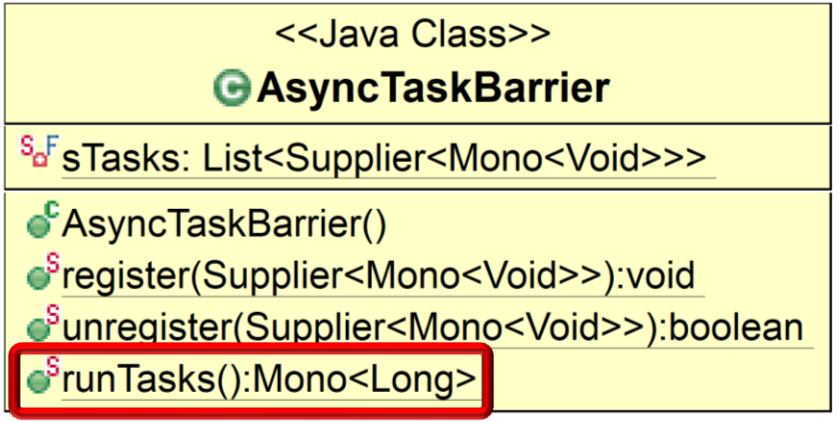
- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks
  - It provides methods that register & unregister tasks passed as Suppliers
    - These tasks are stored in a List



See docs.oracle.com/javase/8/docs/api/java/util/List.html

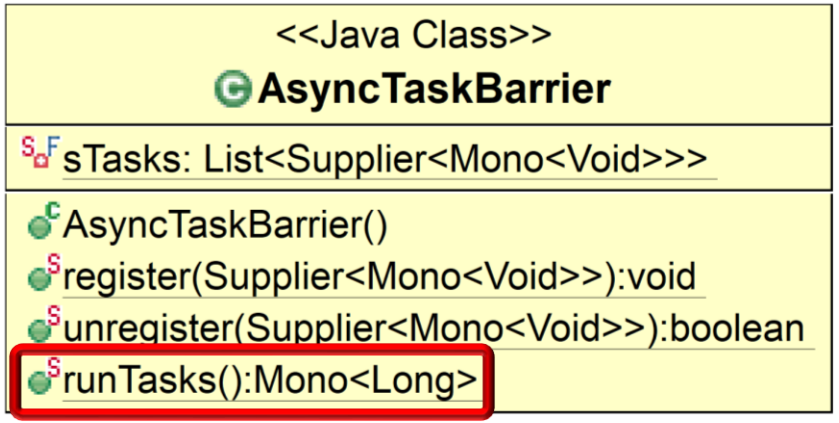
The AsyncTaskBarrier Class API

- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks
  - It provides methods that register & unregister tasks passed as Suppliers
  - It also provides a method that runs all registered tasks (a)synchronously



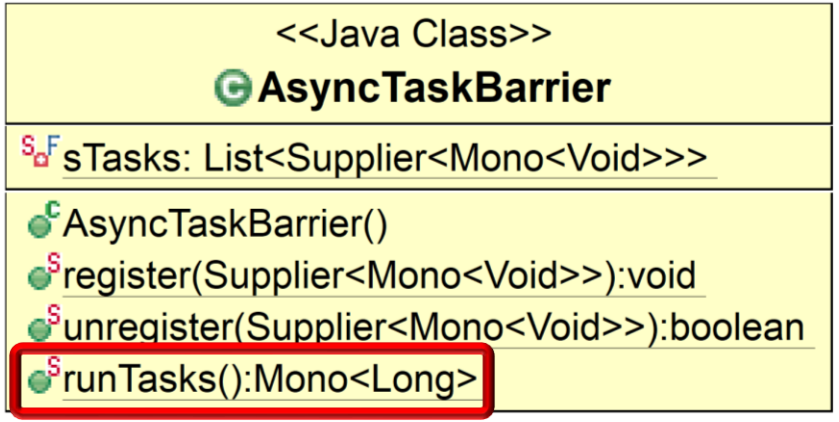
The AsyncTaskBarrier Class API

- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks
  - It provides methods that register & unregister tasks passed as Suppliers
  - It also provides a method that runs all registered tasks (a)synchronously
    - This method doesn't block



The AsyncTaskBarrier Class API

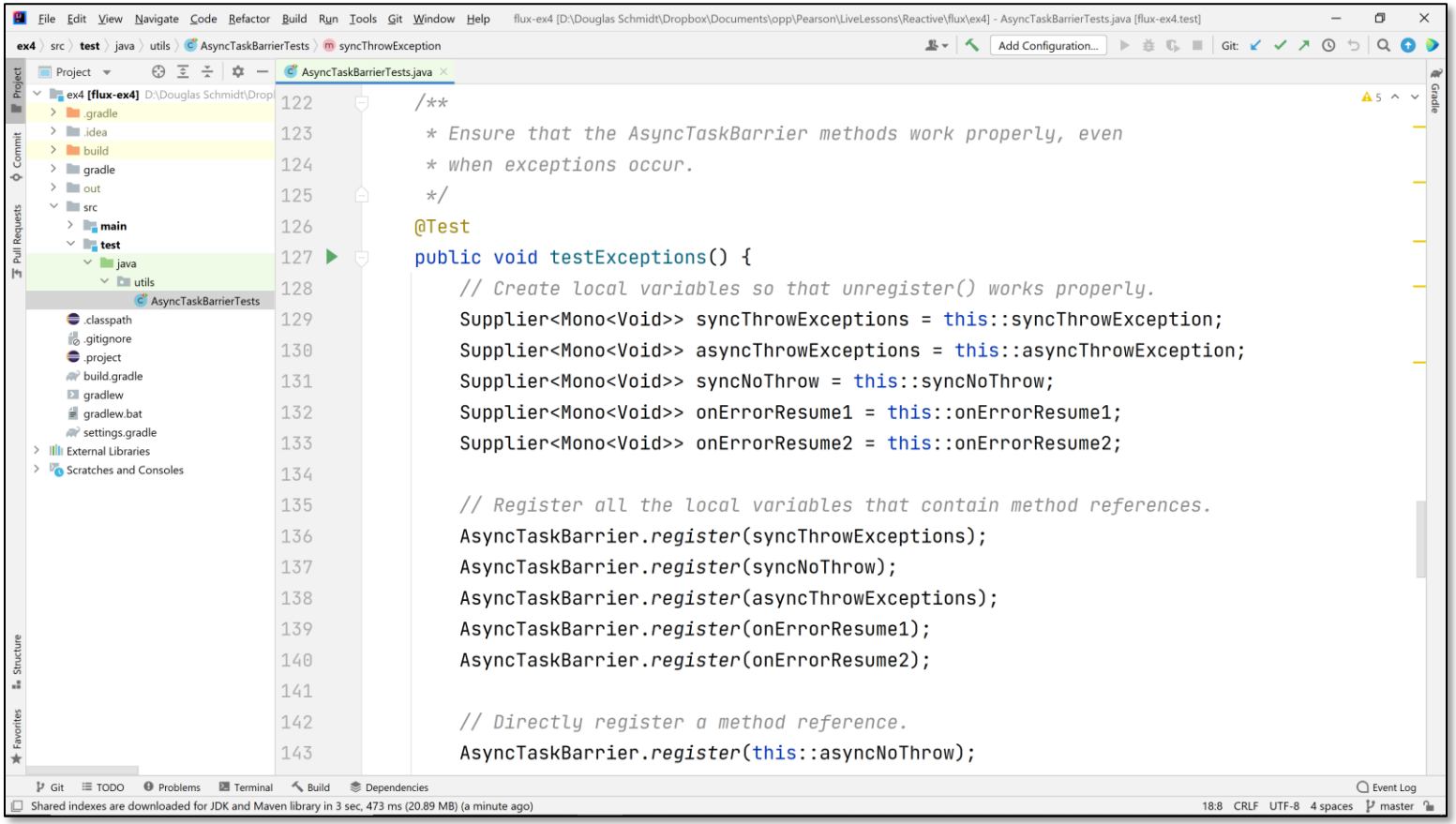
- The AsyncTaskBarrier API contains methods that register, unregister, & (a)synchronously run tasks
  - It provides methods that register & unregister tasks passed as Suppliers
  - It also provides a method that runs all registered tasks (a)synchronously
    - This method doesn't block
    - When combined with Mono.block() the calling thread won't exit until all asynchronous task processing completes



See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#block

Applying the AsyncTask Barrier in Practice

Applying the AsyncTaskBarrier in Practice



See Reactive/flux/ex4/src/test/java/utils/AsyncTaskBarrierTests.java

End of Implementing the AsyncBarrierTask Framework Using Project Reactor (Part 1)